Tokamesh : A software for mesh generation in Tokamaks

Hervé Guillard, Jalal Lakhlili, Alexis Loyer, Adrien Loseille, Ahmed Ratnani, Ali Elarif

▶ To cite this version:

Hervé Guillard, Jalal Lakhlili, Alexis Loyer, Adrien Loseille, Ahmed Ratnani, et al.. Tokamesh : A software for mesh generation in Tokamaks. Renewable Energy meets High Performance Computing: Final Conference of the Energy-Oriented Centre of Excellence, Sep 2018, Nicosia, Cyprus. hal-01946862

HAL Id: hal-01946862

https://hal.inria.fr/hal-01946862

Submitted on 6 Dec 2018

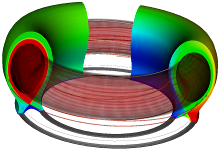
HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Tokamesh : A software for mesh generation in Tokamaks

Hervé Guillard(Inria Sophia), Jalal Lakhlili (IPP), Alexis Loyer (Inria Sophia), Adrien Loseille(Inria Saclay), Ahmed Ratnani(IPP), Ali Elarif (Inria Sophia)

Université Côte d'Azur, Inria, LJAD, CNRS, France

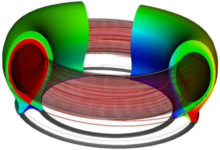


- High anisotropy (10^9) in magnetized fusion plasmas
- Tokamak is an axisymmetric machine :
  $\mathbf{B} = F(\psi)\nabla\phi + \nabla\psi \times \nabla\phi$
  ψ is the magnetic flux

$\Rightarrow$

- use Fourier discretization in the toroidal direction
- use flux aligned meshes in the poloidal direction

EoCoE WP5.1 task: Constructing flux-surface aligned mesh-grids in the poloidal plane



Task WP5.1 : Propose unified software for

- different codes and models
  - gyrokinetic (Gysela)
  - Fluid (Jorek, Tokam3X, SOLPS, SOLEdge, Plato)
- different type of meshes
  - block structured - non structured
- different numerical methods
  - Semi-Lagrangian approaches
  - Finite difference, Finite volume
  - $\mathcal{C}^0$ Finite Element (Lagrange, spectral elements)
  - $\mathcal{C}^1$ Finite Element (Spline or Hermite-Bezier on quadrangles, Powell-Sabin-Clough-Tocher on triangles)

Computing the flux surfaces : Grad-Shafranov Equilibrium solver

The problem

Generate meshes adapted to the magnetic flux surfaces (anisotropy)

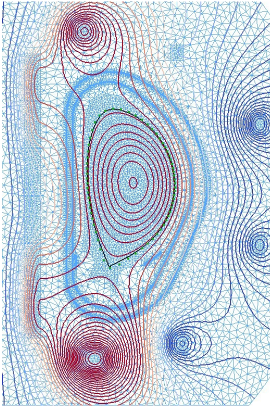
First part of the problem : Compute the magnetic flux surfaces

Magnetic field Ansatz : $\mathbf{B} = F(\psi)\nabla\phi + \nabla\psi \times \nabla\phi$

Force balance $\nabla p = \mathbf{J} \times \mathbf{B}$ and toroidal symmetry yield

Non-linear elliptic free boundary problem :

$$-\nabla\left(\frac{1}{\mu r}\nabla\psi\right) = \mathbf{J}_{\text{toro}} = \begin{cases} \mathbf{J}_{\text{plasma}}(x,\psi)) & \text{in plasma} \\ \mathbf{J}_{\text{coil}}(\text{voltage}, \partial_t\psi)) & \text{in coils} \\ \text{vanishing} & \text{elsewhere} \end{cases}$$



Tokamesh work flow

1. Input from equilibrium codes (EFIT, Cedres++) solving Grad-Shafranov equation
   - get flux field data
2. Automated domain segmentation
   - Decomposition of the domain into patches homeomorphic to a logical square
3. Construction of meshes-grids of individual patches
4. Gluing of patches to obtain the global mesh

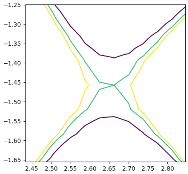
Tokamesh work flow

1. Input from equilibrium code (EFIT, Cedres++) solving Grad-Shafranov equation
   - get flux field data
2. **Data regularization**
3. Automated domain segmentation
   - Decomposition of the domain into patches homeomorphic to a logical square
4. Construction of meshes-grids of individual patches
5. Gluing of patches to obtain the global mesh

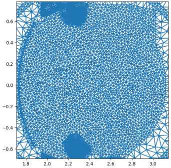
Data regularization

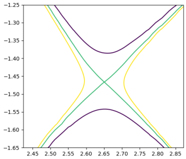
Results from standard equilibrium solvers are not smooth enough

- use of low-order discretization
- coarse resolution

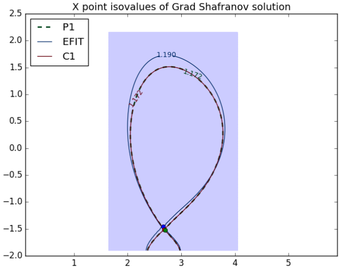


- Mesh refinement around x-points + $\mathcal{C}^1$ Splines interpolation on triangles (Clough-Tocher)





Free boundary Grad-Shafranov Equilibrium



Free boundary Grad-Shafranov Equilibrium solver using Clough-Tocher $\mathcal{C}^1$ finite element method on triangular meshes

- Regularize the data coming from low-order GS solver
- Recompute an equilibrium using $\mathcal{C}^1$ high-order method

Domain decomposition

Segmentation problem :

Given a domain Ω : Find n patches Ω_n such that :

$$\Omega = \cup \Omega_n \quad \text{and} \quad \mathring{\Omega}_n \cap \mathring{\Omega}_m = \emptyset$$

and there exists a one to one mapping between Ω_n and the the unit square :

$$\exists f_n : [0,1]x[0,1] \leftrightarrow \Omega_n$$

In general : extremely difficult question (10^3 of papers on this question)
Additional constraint : we want Flux aligned grids
The problem becomes much easier :

use Morse theory

Morse. M, The Calculus of Variations in the Large, American Mathematical Society Colloquium Publication 18; New York, (1934)

Morse theory

A function f is a Morse function if all its critical points are regular.

Critical points

Let C^r be the space of r differentiable scalar field defined on Ω. For $r \geq 2$, a point $\mathbf{p} \in \Omega$ is a critical or singular point of f if $\nabla f = 0$.

Regular Critical points

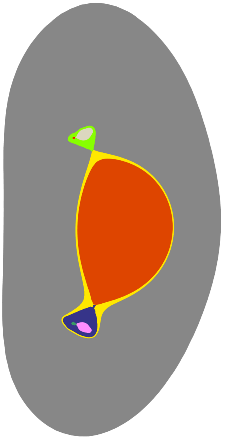
A critical point is regular (or non-degenerate) if the Hessian of f at $\mathbf{p}$ is invertible.

In 2-D

if we note μ_1, μ_2 the two eigenvalues of H_f the only possible critical points of a Morse function are therefore :

Maxima	index $= 2$	$\mu_1 < 0$	$\mu_2 < 0$
Saddles	index $= 1$	$\mu_1 < 0$	$\mu_2 > 0$
Minima	index $= 0$	$\mu_1 > 0$	$\mu_2 > 0$

Morse theory



Mountainer theorem : Let f be a Morse function defined on Ω, a region defined by a closed iso-contour, then the number of critical points (counting a virtual extremum) verifies the relation :

$$C_M - C_S + C_m = 2$$

Topology : Let f be a Morse function defined on Ω. Then the topological set of the iso-contours of f consists of finite connected components that are either

- Circle cells which are homeomorphic to open disks
- Circle bands which are homeomorphic to open annulus
- Saddle connections

Stability : Let f be a Morse function defined on Ω. This field is structurally stable if and only if all saddle points are self-connected.

Morse theory and Reeb Graph

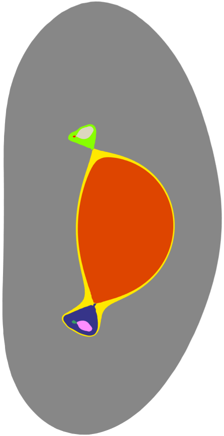


Figure: Domain decomposition

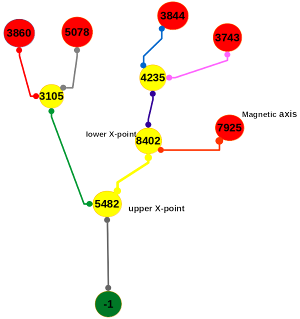
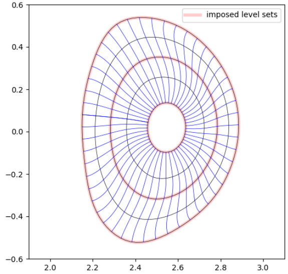


Figure: Associated Reeb graph : Each edge corresponds to a subdomain

Algorithm for block structured mesh generation

Meshing individual patches



Into each subdomain the level set $f^{-1} = c$ is unique and closed

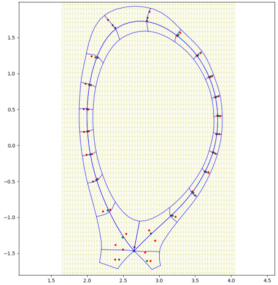
- Choose a finite number of level sets $f^{-1}(c_i), i = 1, \cdots, I$
- Approximate each level set by a cubic spline : $f^{-1}(c_i) \sim \mathcal{C}_i(t) = \sum_j \mathbf{P}^i_j N_j(t)$:
  N_j spline basis function $\mathbf{P}^i_j$: control points
- Construct a 2D tensor product mapping from $[0,1] \times [0,1] \to \Omega_k$
  $\mathcal{S}(s,t) = \sum_{i=1}^{I} \sum_{j=1}^{J} \mathbf{P}_{i,j} N_i(s) N_j(t)$
- such that

$$\forall i = 1, \cdots, I \qquad \mathcal{S}(s_i, t) = \mathcal{C}_i(t)$$

Note that $\mathbf{P}_{i,j} \neq \mathbf{P}^i_j$

Algorithm for block structured mesh generation

$\mathcal{G}^1$ Gluing of patches



Given 2 subdomains Ω_1 and Ω_2 described by 2 mappings

$$\mathcal{S}^1(s,t) = \sum_{i=1}^{I^1} \sum_{j=1}^{J} \mathbf{P}^1_{i,j} N_i(s) N_j(t)$$

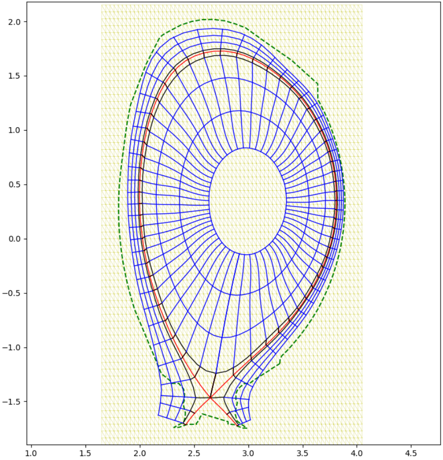
$$\mathcal{S}^2(s,t) = \sum_{i=1}^{I^2} \sum_{j=1}^{J} \mathbf{P}^2_{i,j} N_i(s) N_j(t)$$

with a common boundary for instance at $s = 0$

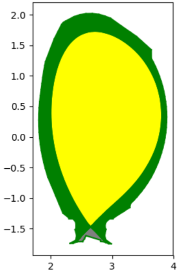
$$\mathcal{S}^1(0,t) = \mathcal{S}^2(0,t)$$

- $\mathcal{G}^0$ continuity : $\mathbf{P}^1_{0,j} = \mathbf{P}^2_{0,j} \qquad \forall j$
- $\mathcal{G}^1$ continuity : geometric condition on the control points $\mathbf{P}^1_{1,j}, \mathbf{P}^1_{0,j} = \mathbf{P}^2_{0,j}, \mathbf{P}^2_{1,j}$ have to be aligned.

$\mathcal{G}^1$ meshes : example

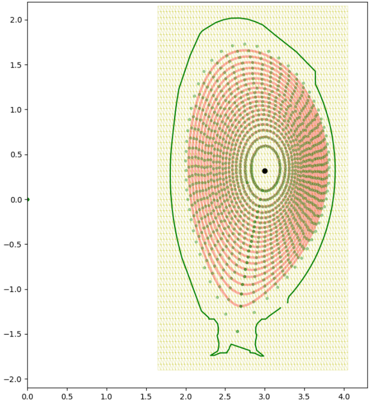


Algorithm for non-structured triangular mesh generation



- Segmentation of the domain
- choice of a set of isolines into each subdomain (depend on user's interest)
- cubic spline approximation of these isolines
- node sampling on each isolines (no need to have the same number of nodes on each isoline) → cloud a nodes
- triangulation of the resulting set of nodes by a constrained anisotropic Delaunay algorithm

Algorithm for non-structured triangular mesh generation



- Segmentation of the domain
- choice of a set of isolines into each subdomain (depend on user's interest)
- cubic spline approximation of these isolines
- node sampling on each isolines (no need to have the same number of nodes on each isoline) $\rightarrow$ cloud a nodes
- triangulation of the resulting set of nodes by a constrained anisotropic Delaunay algorithm

Algorithm for non-structured triangular mesh generation

Constrained Anisotropic Delaunay algorithm

Standard Delaunay algorithm

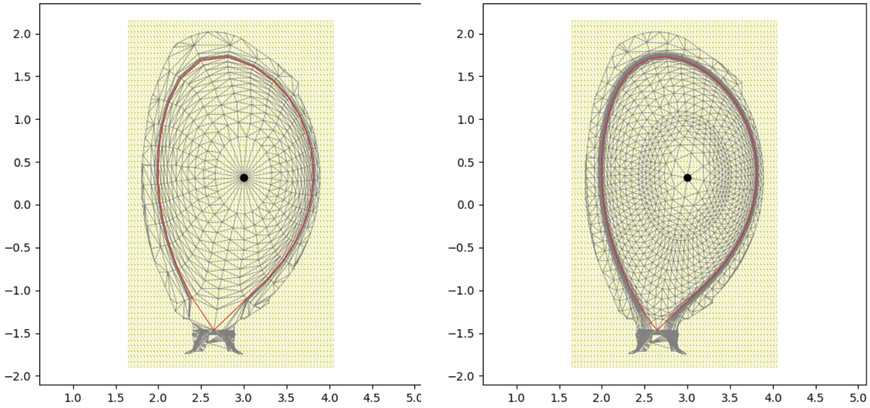
empty sphere property : no point is inside the circumcircle of any triangle

Constrained Anisotropic Delaunay algorithm

Some edges of the triangulation are forced to exist in the triangulation
The distance are evaluated using Riemannian metric instead of Euclidian one

$$d(\mathbf{a}, \mathbf{b}) = \sqrt{(\mathbf{b} - \mathbf{a})^t M^t \Lambda M (\mathbf{b} - \mathbf{a})}$$

Algorithm for non-structured triangular mesh generation



Conclusions

- Design of a mesh generation software for tokamak simulations
  - can be used by different codes
  - handle different types of meshes
  - written in python with C and FORTRAN bindings
    - using open source libraries
    - (Except feflo.a but free for academics)
- future developments
  - GUI
  - extension for unstructured P2- P3 meshes
  - extends the library to geophysical applications.

git@gitlab.inria.fr:EoCoE_Mesh/tokamesh.git